A Parallel Object-Oriented System for Realizing Reusable and Efficient Data Abstractions

Chu-Cheow Lim*

TR-93-063

October 1993

Abstract

We examine the use of an object-oriented language to make programming multiprocessors easier for the general programmer. We choose an object-oriented paradigm because we believe that its support for encapsulation and software reuse allows users who are writing general application programs to reuse class libraries designed by expert library writers.

We describe the design, implementation and use of a parallel object-oriented language: parallel Sather (pSather). PSather has a shared address space independent of the underlying multiprocessor architecture, because we believe that the cooperative nature of parallel programs is most easily captured by a shared-memory-like model. To account for distributed-memory machines, pSather uses an abstract model in which processors are grouped in clusters. Associated with a cluster is a part of the address space with fast access; access to other parts of the address space is ≤ 2 orders of magnitude slower. PSather integrates both control and data-parallel constructs to support a variety of algorithmic styles.

We have an implementation of pSather on the CM-5. The prototype shows that even on distributed-memory machines without hardware/operating system support for a shared address space, it is still practical and reasonably efficient for the shared address abstraction to be implemented in the compiler/runtime. The experience also helps us understand the features of low-level libraries that are necessary for an efficient realization of a high-level language. For example, even though low message latency is crucial, the message-passing paradigm (active vs. passive, polling vs. interrupt-driven) is also important in deciding how easy and efficient the language

*ICSI and Computer Science Division, U.C. Berkeley. E-mail clim@icsi.berkeley.edu.

implementation will be. We also study certain straight-forward compiler optimizations.

Several abstractions and applications have been written for the CM-5 using the shared-address cluster model, and we have achieved reasonable speedups. In some cases, we can further demonstrate good absolute performance for pSather programs (by getting their speedups relative to a 1-processor C program). Some of the abstractions are reused in several applications, to show how the object-oriented constructs facilitate code reuse.

The work described here supports our optimism that pSather is a practical and efficient parallel object-oriented language. There are, however, still many issues that need to be explored in order to provide parallel programming environments as powerful as the ones we are accustomed to on sequential environments. In the conclusion, we summarize some of the possible future research directions.